Remarks

Reconsideration of this Application is respectfully requested.

Upon entry of the foregoing amendment, claims 1-73 are pending in the application, with 1, 27, 40, 53 and 67- 71 being the independent claims. Claims 1, 27, 40, 53, 67, 68, 69, 70 and 71 are amended. These changes are believed to introduce no new matter, and their entry is respectfully requested.

In the Final Office Action dated January 22, 2007, the Amendment filed on November 9, 2006 is objected to under 35 U.S.C. § 132(a). Claims 1-3, 6, 7, 13-17, 19-29, 31, 32, 35-45, 48, 49, 51-55, 57, 61-64 and 66-73 stand rejected under 35 U.S.C. § 102(b) as being allegedly anticipated by Mahalingaiah et al., U.S. Patent No. 5,983,337. Claims 5, 8, 10, 11, 12, 30, 33, 34, 46, 47, 56, 59 and 60 stand rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Mahalingaiah. Claims 4, 18 and 50 stand rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Mahalingaiah in view of Scott et al., U.S. Patent No. 6,615,329.

Based on the above amendment and the following remarks, Applicants respectfully request that the Examiner reconsider all outstanding objections and rejections and that they be withdrawn.

Interview at USPTO on February 27, 2007

Applicant's representative thanks the Examiner for the courtesies extended during the in-person interview at the USPTO. As discussed further below, and as discussed in detail during the interview, Applicants believe that patching of microcode, and on-the-fly patching of

processor instructions, to which this application is directed, are completely different things, and, in view of the claim language, the pending claims cannot read on the disclosure of Mahalingaiah. Additionally, the Examiner requested during the interview that the term "mark instructions" be further clarified in the claims. Applicants have done so, and note that the language added to all the independent claims to clarify the meaning of "mark instructions" is directly from the definition in the specification, see paragraph 34 of the specification. Thus, applicants respectfully submit that, since the amended language does not in any way change the scope of the claims, no new search and/or consideration is required.

Objection to Amendment Filed on November 9, 2006

The Amendment in response to the first Office Action is objected to due to allegedly incorporating new matter. As discussed during the interview on February 27, 2007, the language of the Amendment relating to "instructions that are part of the instruction set of the processor available to a user" are fully supported in the specification. For example, the specification, at paragraph 28 refers to:

[0028] "Code" generally refers to binary or machine code."

Also, at paragraph 59, there is a discussion of the Intel X86 architecture – it is clear to one of ordinary skill in the art that the reference is that is what is being discussed there is the Intel X86 instruction set, which is obviously available to a user to program the microprocessor (either in binary form, or using assembly language mnemonics, such as MOV, LGDT, XCHG, PUSHF, etc.). Also, various examples of such instructions are given throughout the specification

(e.g., "pushad" and "push esp"), which again illustrate an example of instructions from a processor's instruction set (in this example, the Intel X86 instruction set), that are available to a user.

Also, paragraphs 65 and 66, also reproduced below, give several examples of such instructions.

In sum, Applicant respectfully submits that the claim language is fully supported by the specification, and request that the objection be withdrawn.

Rejections under 35 U.S.C. § 102(b) and 103(a)

All of the claims stand rejected based on Mahalingaiah, or Mahalingaiah in combination with other references. These rejections are respectfully traversed.

As discussed at length during the interview at the USPTO, Mahalingaiah is directed to **patching of microcode** while the present application is directed to **patching of executable (machine, or binary) code** – these are in no way the same thing.

Specifically, as a result of following the approach of Mahalingaiah, the behavior of the CPU itself is changed – by changing the contents of the microcode read-only memory (MROM), the behavior of a particular instruction will therefore be different. This might have some commercial relevance (in view of AMD, the owner of the Mahalingaiah patent), particularly with relatively complex instructions, such as those that take a large number of clock cycles to complete.

One example discussed during the in-person interview at the USPTO is the behavior of the load global descriptor table (LGDT) instruction, which in the Intel processor, at certain points during execution, where an exception is raised, can behave in a somewhat peculiar manner (in essence, due to what can be viewed as a hardware bug, the behavior of this instruction, in some specific circumstances, is not what logic would suggest it should be). Thus, Mahalingaiah proposes a scheme where the behavior of the processor itself, when it executes such an instruction, would be changed. The relevant point here is that Mahalingaiah changes the properties of the microprocessor itself – not the code that is being executed.

Furthermore, microcode instructions are not available to a user – the microcode cache (MROM) cannot be programmed to execute any "useful" user code – it is only programmed to define the behavior of some processor instructions – load registers, push flags, and so forth.

On the other hand, the present application is directed to changing the behavior of the user code (executable code, or machine code) that is loaded into memory, and which is being executed by the processor. This user code is obviously, when loaded into memory, in the form of binary code (machine code), although to simplify human readability, the binary (machine) code, is often written using various mnemonics, such as MOV, PUSHF, PUSHA, LGDT, LLDT, and so forth. As discussed during the interview, these representations are conceptually interchangeable, although it is understood that what is actually loaded into memory is object code in binary form, rather than assembly language letter-type mnemonics.

Unlike Mahalingaiah, the present application is not directed to changing the behavior of the microprocessor (in a certain sense, Mahalingaiah may be viewed as "rewiring" the

microprocessor itself, to achieve different behavior). In the present application, the processor does not behave any differently from before – all the instructions continue to execute in exactly the same manner as they did before. The only question is: which instructions are going to be executed.

The present application is therefore directed to replacing, or patching, instructions from the instruction set of the processor (for example, the X86 instruction set) with other instructions. For example, as one example, the instruction MOV EAX, ECX (see paragraphs 65-66 of the specification) may be replaced by the instruction PUSHF (push flags down onto the processor stack – an instruction whose number of executions can be counted), with a subsequent jump to another location.

Thus, in view of the above discussion, Applicants believe that the pending claims clearly distinguish over the microcode patching of Mahalingaiah. Specifically, using claim 1 as an example, this claim recites that "the original instructions are part of the instruction set of the processor available to a user." Examples of these instructions are given in, for example, paragraphs 65 and 66 as follows:

> [0065] First, a conventional "dangerous" method of patching code on the fly will be described. It is assumed that the reader is familiar with the Intel X86 architecture and its instruction set. In this patch, the first five bytes are copied to a stub using the jump/call instruction.

```
Was:
0000: B8 CC EB 04 CC        mov eax,0xCC04EBCC

Becomes:
0000: EB ZZ ZZ ZZ ZZ        jmp stub_code
```

[0066] A special case of these five bytes is where the instructions are "one within another", for example, used in the case of copy protection is as follows:

```
0000: B8 CC EB 04 CC        mov eax,0xCC04EBCC
[002]: EB 04                jmp $+6                 //to 008:
0005: EB FB                 jmp $-3                 //to 002:
0007:
0008:
```

```
Was: (fastcall function return sum of two numbers)
0000: 8B C1                 mov eax,ecx
0002: 03  C2                add eax,edx
0004: C3                    retn

Becomes:
0000: EB ZZ ZZ ZZ ZZ        jmp stub_code
[0002:] ZZ ZZ ZZ
```

As discussed above, these instructions are not what Mahalingaiah is about – Mahalingaiah has nothing to do with replacing instructions of this type – Mahalingaiah replaces **microcode** instructions. Microcode instructions are used as building blocks to implement an instruction of the processor instruction set, and microcode instructions are not available to a user to program the processor.

Furthermore, **as a separate ground for patentability**, claim 1, as amended, recites the use of "**mark instructions**." These are defined in paragraph 34 as follows:

> [0034] A "mark instruction" refers to tags, or instructions that leave "countable" marks somewhere, that can then be counted, but without performing any other operations that affect the state of the program.

By way of illustration, after executing a sequence of the type:

MOV EAX, 0

MOV EAX, 0

MOV EAX, 0

…

…

MOV EAX, 0

it cannot be determined how many times such an instruction has been executed.

Mahalingaiah does not have anything of the sort. Mahalingaiah mentions "marked instructions," in column 5, lines 53-55, and column 26, lines 23-25. The microcode "marked instructions" of Mahalingaiah, and the "mark instructions" of the present application have nothing in common, except a superficial semantic similarity. They certainly change the state of the processor – and – equally important – the number of times they are executed cannot be counted.

In any event, in view of the amended language, Applicants respectfully submit that the claimed language unambiguously distinguishes over Mahalingaiah.

Reconsideration and allowance of this application is respectfully requested.

Conclusion

All of the stated grounds of objection and rejection have been properly traversed, accommodated, or rendered moot. Applicants therefore respectfully request that the Examiner reconsider all presently outstanding objections and rejections and that they be withdrawn. Applicants believe that a full and complete reply has been made to the outstanding Office Action and, as such, the present application is in condition for allowance. If the Examiner believes, for any reason, that personal communication will expedite prosecution of this application, the Examiner is invited to telephone the undersigned at the number provided.

Prompt and favorable consideration of this Amendment and Reply is respectfully requested.

Respectfully submitted,

BARDMESSER LAW GROUP

/GB/

George S. Bardmesser
Attorney for Applicants
Registration No. 44,020

Date: March 5, 2007

910 17th Street, N.W.
Suite 800
Washington, D.C. 20006
(202) 293-1191